

## Task 01 (Graph Traversal Using BFS):

You have most probably heard of Dora The Explorer. She likes to travel from one country to another.

Currently, Dora is at Doraland. Doraland is a beautiful country, consisting of  $N$  cities and  $M$  **bidirectional** roads.

Recently, Dora has started to learn Graph Algorithms. She knows about BFS and DFS. However, since Dora is still learning, she is not feeling so confident and asks for your help.

Dora contacts you and gives you all the necessary information about Doraland. **Dora will start her journey from city 1.** Now, your task will be to help Dora to find a path which is similar to the path of BFS graph traversal.

### Input

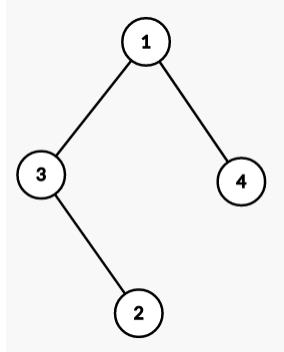
**The given map will be an undirected and unweighted graph.**

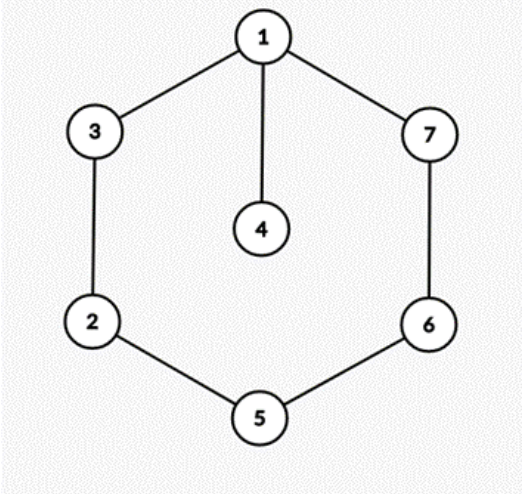
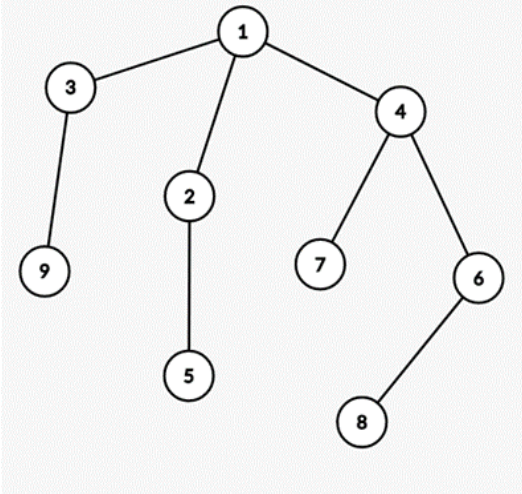
The first line contains two integers  $N$  and  $M$  ( $1 \leq N, M \leq 100$ ) – the number of cities and the total number of roads.

The next  $M$  lines will contain two integers  $u_i, v_i$  ( $1 \leq u_i, v_i \leq N$ ) – denoting there is a bidirectional road between  $u_i$  and  $v_i$ .

### Output

Print the BFS path traversal which Dora will follow to explore the city, starting from city 1.

Sample Input 1	Sample Output 1	Sample Graph 1
4 3 1 3 3 2 1 4	1 3 4 2  ( Another valid path: 1 4 3 2 )	
Sample Input 2	Sample Output 2	Sample Graph 2

<pre> 7 7 1 3 3 2 1 4 2 5 5 6 1 7 7 6 </pre>	<pre> 1 3 4 7 2 6 5 </pre>	
Sample Input 3	Sample Output 3	Sample Graph 3
<pre> 9 8 1 2 1 3 1 4 2 5 4 6 4 7 6 8 3 9 </pre>	<pre> 1 2 3 4 5 9 6 7 8 </pre>	

**This problem may have multiple valid solutions, however your solution should follow the BFS level order traversal.**

#### **Pseudocode of BFS:**

The breadth-first-search procedure BFS below assumes that the input graph  $G = (V, E)$  is represented using adjacency lists.

```

BFS(G,s):
1 for each vertex u in G.V:
2     u.colour = 0
3 Q = ∅ ;
4 s.colour = 1
5 ENQUEUE(Q,s)
6 while Q ≠ ∅;
7     u = DEQUEUE(Q)
8     for each v in G.Adj[u]:
9         if v.colour == 0:
10             v.colour = 1
11             ENQUEUE(Q,v)

```

## **Task 02 (Graph Traversal Using DFS):**

After successful BFS traversal, Dora wants to visit the whole country again. However, this time Dora wants to traverse the country following the path which is similar to the path of DFS graph traversal.

Yes, you guessed it correctly. You have to help Dora this time also.

The input format remains the same as Task 02. **Dora will start her journey from city 1 again.**

### **Input**

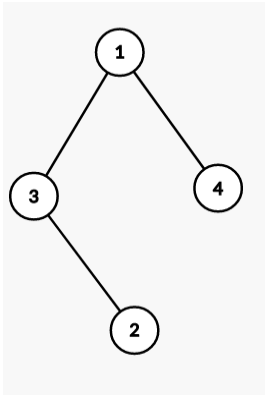
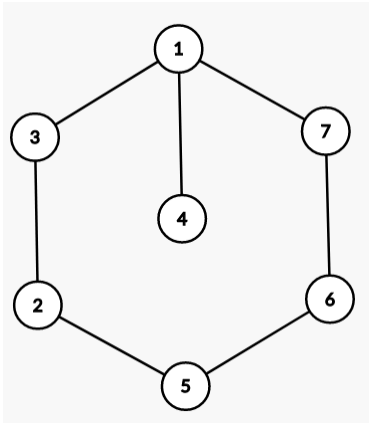
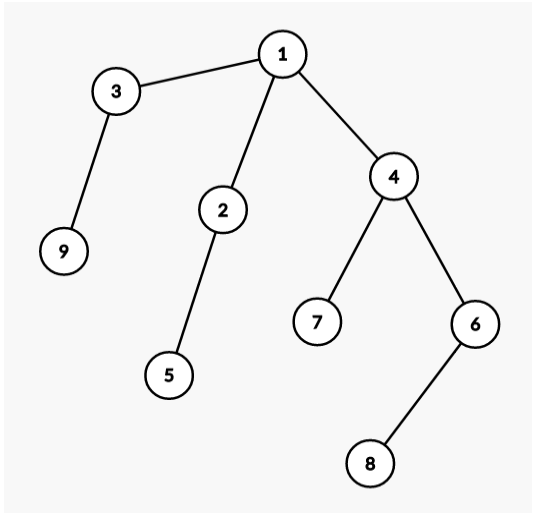
**The given map will be an undirected and unweighted graph.**

The first line contains two integers N and M ( $1 \leq N, M \leq 100$ ) – the number of cities and the total number of roads.

The next M lines will contain two integers  $u_i, v_i$  ( $1 \leq u_i, v_i \leq N$ ) – denoting there is a bidirectional road between  $u_i$  and  $v_i$ .

### **Output**

Print the DFS path traversal which Dora will follow to explore the city, starting from city 1.

Sample Input 1	Sample Output 1	Sample Graph 1
4 3 1 3 3 2 1 4	1 3 2 4  ( Another valid path: 1 4 3 2 )	
Sample Input 2	Sample Output 2	Sample Graph 2
7 7 1 3 3 2 1 4 2 5 5 6 1 7 7 6	1 3 2 5 6 7 4	
Sample Input 3	Sample Output 3	Sample Graph 3
9 8 1 2 1 3 1 4 2 5 4 6 4 7 6 8 3 9	1 2 5 3 9 4 6 8 7	

This problem may have multiple valid solutions. However, your solution should follow the dfs path traversal order.

### Pseudocode of DFS:

The depth-first-search procedure DFS below assumes that the input graph  $G = (V, E)$  is represented using adjacency lists.

```
colourInitializing(G):
1 for each vertex u in G.V:
2     u.colour = 0

DFS(G,u):
1     u.colour = 1
2     for each v in G.Adj[u]:
3         if v.colour == 0:
4             DFS(G,v)
```

### Task 03 (Cycle Finding):

It was a very hectic day for Dora. Before going to sleep, Dora wants to find out if there is any Cycle in the map of the city.

Since Dora has traveled the whole country twice, she is feeling very exhausted and asks you to solve the problem. However, Dora has made the roads of the cities **directed** in her map, so that you don't get bored.

In graph theory, a cycle in a graph is a non-empty trail in which only the first and last vertices are equal.

#### **Input:**

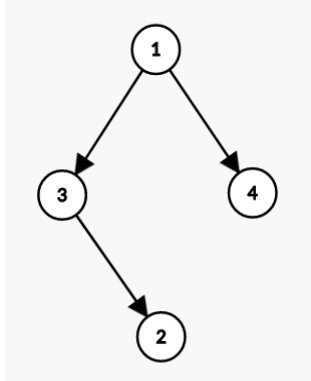
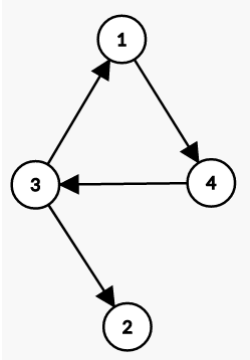
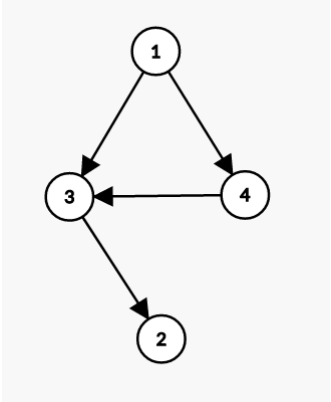
**The given map will be a directed and unweighted graph.**

The first line contains two integers  $N$  and  $M$  ( $1 \leq N, M \leq 100$ ) – the number of cities and the total number of roads.

The next  $M$  lines will contain two integers  $u_i, v_i$  ( $1 \leq u_i, v_i \leq N$ ) – denoting there is a bidirectional road between  $u_i$  and  $v_i$ .

#### **Output:**

Print "YES" if the map contains any Cycle, otherwise print "NO".

Sample Input 1	Sample Output 1	Sample Graph 1
4 3 1 3 3 2 1 4	NO	 <pre> graph TD     1((1)) --&gt; 3((3))     1((1)) --&gt; 4((4))     3((3)) --&gt; 2((2)) </pre>
Sample Input 2	Sample Output 2	Sample Graph 2
4 4 3 1 3 2 1 4 4 3	YES	 <pre> graph TD     1((1)) --&gt; 3((3))     1((1)) --&gt; 4((4))     3((3)) --&gt; 2((2))     4((4)) --&gt; 3((3)) </pre>
Sample Input 3	Sample Output 3	Sample Graph 3
4 4 1 3 3 2 1 4 4 3	NO	 <pre> graph TD     1((1)) --&gt; 3((3))     1((1)) --&gt; 4((4))     3((3)) --&gt; 2((2))     4((4)) --&gt; 1((1)) </pre>

#### **Task 04 (Find the shortest path):**

The next day, Dora again called you for helping her. Among all the cities she likes city 'D' the most.

Since you are becoming very much annoyed with Dora, you have planned to reach city D spending the minimum amount of time.

In the belugaland, moving between two cities connected by a road takes one second. You have to find the minimum time to reach D and show the path in the output.

Dora and you will start your journey from city 1.

#### **Input**

**The given map will be an undirected and unweighted graph.**

The first line contains three integers N, M, and D ( $1 \leq N, M, D \leq 100$ ) – the number of cities, the total number of roads and the destination city.

The next M lines will contain two integers  $u_i, v_i$  ( $1 \leq u_i, v_i \leq N$ ) – denoting there is a bidirectional road between  $u_i$  and  $v_i$ .

#### **Output**

Print the minimum amount of time to reach city D from city 1.

Next, print the shortest path you will follow.

Sample Input 1	Sample Output 1
4 3 2 1 3 3 2 1 4	Time: 2 Shortest Path: 1 3 2
Sample Input 2	Sample Output 2
6 6 5 1 3 3 2 1 4 2 6 5 6 4 6	Time: 3 Shortest Path: 1 4 6 5

Sample Input 3	Sample Output 3
9 8 7 1 2 1 3 1 4 2 5 4 6 4 7 6 8 3 9	Time: 2 Shortest Path: 1 4 7
Sample Input 4	Sample Output 4
11 14 5 4 2 1 3 10 4 2 5 4 6 4 7 6 8 3 9 3 7 1 8 4 8 1 10 9 11 10 11	Time: 4 Shortest Path: 1 8 4 2 5
Sample Input 5	Sample Output 5
4 3 1 1 3 3 2 1 4	Time: 0 Shortest Path: 1

**This problem may have multiple valid shortest paths. However, the minimum time required to reach the destination should be matched exactly with the output.**

**Bonus :**

**Task 01:**



Dora is leaving the country. Before leaving the country, as a token of gesture, Dora gifted you with a map of the jungle "Jumanji".

The map is very interesting. The map is a 2D grid. Some of those cells are occupied by diamonds and obstacles. [See the Sample input for better understanding.]

Now, you are on a journey to Jumanji in search of Diamonds. However, since the jungle is full of ferocious creatures, you can't collect all the diamonds. **You will choose only one start position such that you collect the maximum amount of diamonds. Please note that you can not move to any cell which contains obstacles.**

### Input

The first line contains two integers R and H ( $1 \leq R, H \leq 100$ ) – the number of rows and columns respectively in the grid.

The next R line will contain H characters. Each character represents the status of a cell as follows.

- 1) '.': Empty Cell → You can move here.
- 2) 'D': Cell with a Diamond → If you move to the cell containing 'D', you will collect that Diamond.
- 3) '#': Cell with an obstacle → You can't move to this cell.

### Output

Print a single integer that denotes the maximum amount of Diamonds you can collect.

**[The diamonds are coloured red to demonstrate which diamonds have been collected.]**

Sample Input 1	Sample Output 1
4 3 ..D D.. .D. ##.	3
Sample Input 2	Sample Output 2

10 7 ...#...D ...#D.. D...#.D. D...#... DDD#### ..... .####.. .#D.#DD .####.. DDD...D	11
Sample Input 3	Sample Output 3
9 11 .#...D...D.. .#.#####. D#.#...D...#. D#D#.#...#D .#.#...D#.#. .#.#####.#D D#...D...D#. .#####. ...D..D...D	15
Sample Input 4	Sample Output 4
5 5 ..... ####. #D.#. ####. .....	1
Sample Input 5	Sample Output 5
5 5 ..... ####. #..#. ####. .....	0
Sample Input 6	Sample Output 6
1 5 D.....	1
Sample Input 7	Sample Output 7
12 12 .....	4

<pre> .####..... .#D.#..... .####..... .....###. ...D....#D#. .....#D#. #####D#. .#D....##.#. .#.D..D##D#. #####. ..... </pre>	
--	--

## Task 2:

Dora returns to Doraland once again. She finds that the city has undergone significant improvements since her last visit. The city structure has been reconstructed to make it more beautiful. Currently, **Doraland consists of N cities and all the cities are connected with N-1 bidirectional roads.**

However, this time, Dora will have a very short trip and as usual, she seeks your help. According to Dora's wish, you will choose two cities, let's say city A and city B in such a way that while traveling from city A to city B Dora can pass through a maximum number of cities. You have to keep in mind that it is not possible to visit any roads twice in the journey since Dora is on a short trip.

Can you help Dora find such two cities?

## Input

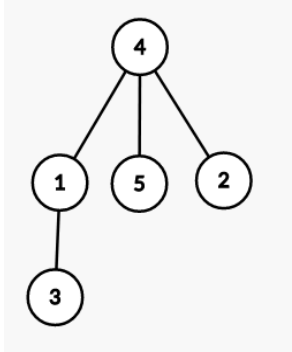
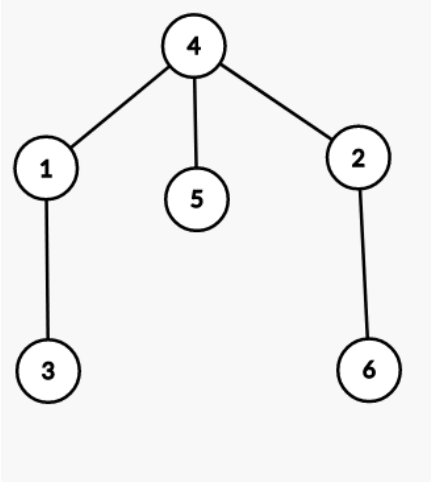
**The given map will be an undirected and unweighted graph.**

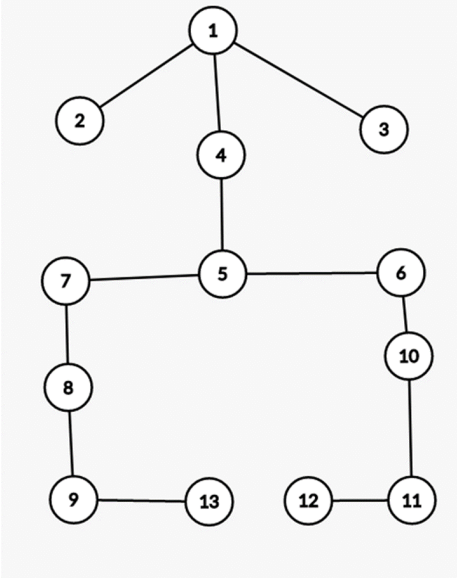
The first line contains one integer N ( $2 \leq N \leq 100$ ) – the number of cities.

The next N-1 lines will contain two integers  $u_i, v_i$  ( $1 \leq u_i, v_i \leq N$ ) – denoting there is a bidirectional road between  $u_i$  and  $v_i$ .

## Output

Print two cities A and B which satisfy the conditions stated in the problem statement.

Sample Input 1	Sample Output 1	Sample Graph 1
5 4 1 4 2 1 3 4 5	2 3  Or ( 3 5 )  [You may print only one valid output]	 <pre> graph TD     4((4)) --- 1((1))     4 --- 5((5))     4 --- 2((2))     1 --- 3((3)) </pre>
Sample Input 2	Sample Output 2	Sample Graph 2
6 4 1 4 2 1 3 4 5 2 6	3 6	 <pre> graph TD     4((4)) --- 1((1))     4 --- 5((5))     4 --- 2((2))     1 --- 3((3))     2 --- 6((6)) </pre>
Sample Input 3	Sample Output 3	Sample Graph 3

13 1 2 1 3 1 4 4 5 5 6 5 7 7 8 8 9 9 13 6 10 10 11 11 12	12 13	
--	-------	---

**This problem may have multiple valid answers. However, make sure it satisfies the given condition.**

#### Explanation:

In test case 1, if you choose city 2 and city 3, then the path will be

$2 \rightarrow 4 \rightarrow 1 \rightarrow 3$ . If you choose city 3 and city 5, then the path will be

$3 \rightarrow 1 \rightarrow 4 \rightarrow 5$ .

Since, in both paths you can visit a maximum of four cities by using the roads only one, both the answers are valid.

In test case 2, one can see that visiting from city 3 to city 6 ( or vise-versa) is only the valid answer. The path will be  $3 \rightarrow 1 \rightarrow 4 \rightarrow 2 \rightarrow 6$ .